This listing of claims will replace all prior versions, and listings, of claims in the application.

IN THE CLAIMS

1. (Currently Amended) A floating point execution unit for performing multiply/add operations on a floating point number comprising a plurality of operands taken from an instruction having a plurality of floating point number operand positions, the floating point unit comprising:

a multiplier for calculating a product of two of the operands;

a first data path for supplying to the multiplier a first and second of the operands from the instruction, wherein the multiplier multiplies said first and second operands to produce a product;

an aligner ~~directly coupled to the multiplier for aligning said product and a third of the operands in a first data path~~;

~~wherein the first data path is for supplying to the multiplier operands from a first and a second of the operand positions of the instruction;~~

a second data path for supplying ~~the third operand~~ one of the operands from the instruction to the aligner; [[and]]

a multiplexer on the second data path for ~~selecting, for use by the aligner, either the operand from the second operand position of the instruction or the operand from the third operand position of the instruction, and supplying same to the multiplier~~ receiving the second operand and a third of the operands from the instruction, for selecting one of said second and third operands, and sending said selected one of the operands to the aligner;

a shift calculator for determining a shift amount for said selected one of the operands, including a first unit for calculating a shift component using exponents of the first and second operands, and a second unit for calculating said shift amount using said shift component and an exponent and an exponent of the selected one of the operands, wherein said aligner shifts the selected one of the operands by said shift amount to generate a shifted operand; and

an adder for adding the product from the multiplier and the shifted operand to produce a result; and

wherein the first data path is maintained free of multiplexer operations~~, thereby increasing a speed at which the floating point number is output~~.

Claims 2-5 (Cancelled).

6. (Previously Presented) A floating point execution unit according to Claim 1, wherein each of the operands has an exponent value, and floating point execution unit determines whether the exponent values of any of the operands is zero in parallel with operation of the multiplier and the aligner.

7. (Previously Presented) A floating point execution unit according to Claim 6, wherein said floating point execution unit tests said exponent values for a zero value while the multiplier calculates said product.

8. (Previously Presented) A floating point execution unit according to Claim 6, wherein the floating point execution unit establishes a test result number based on results of said

determination.

Claim 9 (Cancelled).

10. (Previously Presented) A floating point execution unit for performing multiply/add operations on a floating point number comprising a plurality of operands taken from an instruction having a plurality of floating point number operand positions, the floating point unit comprising:

a multiplier for calculating a product of two of the operands;

an aligner directly coupled to the multiplier for aligning said product and a third of the operands in a first data path;

wherein the first data path is for supplying to the multiplier operands from a first and a second of the operand positions of the instruction;

a second data path for supplying the third operand to the aligner; and

a multiplexer on the second data path for selecting, for use by the aligner, either the operand from the second operand position of the instruction or the operand from the third operand position of the instruction, and supplying same to the multiplier;

wherein the first data path is maintained free of multiplexer operations, thereby increasing a speed at which the floating point number is output;

wherein each of the operands has an exponent value, and the floating point execution unit determines whether the exponent values of any of the operands is zero in parallel with operation of the multiplier and the aligner, and the floating point execution unit establishes a test result number based on results of said determination;

wherein the test result number includes a plurality of bits, a first of the bits indicates whether the addend is zero, and a second of the bits indicates whether the product is zero; and

wherein the plurality of bits are used to force special values into the aligner result.

11. (Currently Amended) A floating point execution unit according to Claim [[3]] 1, wherein the aligner compresses two of the three input exponents and an offset while selecting the third exponent.

12. (Previously Presented) A floating point execution unit according to Claim 11, wherein, when executing an add or subtract instruction, the aligner computes the alignment shift amount as: [exponent ea + exponent eb – exponent 2eb].

13. (Currently Amended) A method of operating a floating point execution unit to perform multiply/add operations on a floating point number, the floating point unit having a multiplier, an aligner directly coupled to the multiplier in a first data path, and a multiplexer, the method comprising steps:

sending an instruction to the floating point unit, including a floating point number upon which multiply/add operations are to be performed, the instruction having a plurality of operand positions holding operands of the floating point number;

using the multiplier to calculate a product of two of the operands of the instruction;

supplying to the multiplier over a first data path a first and second of the operands from the instruction, wherein the multiplier multiplies said first and second operands to produce a product;

using the aligner to align said product and a third of the operands of the instruction;

~~supplying over the first data path to the multiplier operands from a first and a second of the operand positions of the instruction, wherein said first data path is free of multiplexer operations;~~

supplying over a second data path ~~a third operand of~~ one of the operands from the instruction to [[the]] an aligner;

~~positioning the~~ using a multiplexer on the second data path for receiving the second operand and a third of the operands from the instruction, for selecting one of said second and third operands, and sending said selected one of the operands to the aligner;

~~using the multiplexer to select, for use by the aligner, either the operand from the second operand position or the operand from the third operand position;~~

~~outputting the selected operands to the aligner; and~~

~~outputting the floating point number upon which the multiply/add operations were performed;~~

determining a shift amount for said selected one of the operands, including calculating a shift component using exponents of the first and second operands, and calculating said shift amount using said shift component and an exponent of the selected one of the operands;

said aligner shifting the selected one of the operands by said shift amount to generate a shifted operand; and

adding the product from the multiplier and the shifted operand to produce a result; and

wherein the first data path is maintained free of multiplexer operations~~, thereby increasing a speed at which the floating point number is output~~.

Claims 14-17 (Cancelled).

18. (Original) A method according to Claim 13, wherein each of the operands has an exponent value, and comprising the further step of, determining, in parallel with the multiplier and the aligner, whether the exponent values of any of the operands is zero.

19. (Original) A method according to Claim 18, wherein the step of determining whether the exponent values of any of the operands is zero occurs while the multiplier calculates said product.

20. (Original) A method according to Claim 18, comprising the further steps of: establishing a test result number based on results of said determination, the test result number including a plurality of bits;

using a first of the bits to indicate whether the addend is zero; and

using a second of the bits to indicate whether the product is zero.

21. (Previously Presented) The floating point execution unit according to Claim 1, wherein operand muxing occurs in the aligner and the operand muxing in the aligner is merged with a shift-amount calculation.

22. (New) The floating point execution unit according to Claim 1, wherein the shift calculator further includes a multiplier for receiving exponents of the second and third operands, and for selecting the exponent of one of said second and third operands, and sending the selected

exponent to said second unit.

23. (New) The floating point execution unit according to Claim 22, wherein the multiplier of the shift calculator performs the selecting of the exponent of one of said second and third operands while the first unit of the shift calculator calculates the shift component.